

Ved Babar

Ved Babar

## Ved Babar

Test ID: 450011492000261 | 8830733294 | vedbabar12345@gmail.com

Test Date: April 23, 2025

### Critical Reasoning

58 /100



### C++ Programming

50 /100



### Quantitative Ability

79 /100



### English Comprehension

63 /100



### Logical Ability

61 /100



### Automata

100 /100



### Managerial In-basket Simulation

50 /100



## C++ Programming

50 / 100

### Fundamentals of C++

75 / 100

### OOPS Concepts

50 / 100

### File and Exception Handling

25 / 100

## Quantitative Ability

79 / 100

### Number Theory

78 / 100

### Basic Numbers

76 / 100

### Applied Mathematics

84 / 100

## English Comprehension

63 / 100

CEFR: B2

### Vocabulary

60 / 100

### Grammar

67 / 100

### Comprehension

61 / 100

## Logical Ability

61 / 100

Inductive Reasoning

Deductive Reasoning

58 / 100

65 / 100

## Automata

100 / 100

Programming Practices

Functional Correctness

100 / 100

100 / 100

## Managerial In-basket Simulation

50 / 100

Effective Communication

Planning and Scheduling

Process Adherence

57 / 100

0 / 100

56 / 100

Prioritization

Work Allocation

71 / 100

68 / 100

There are no subsections for the following module(s): Critical Reasoning

## 1 | Introduction

### About the Report

This report provides a detailed analysis of the candidate's performance on different assessments. The tests for this job role were decided based on job analysis, O\*Net taxonomy mapping and/or criterion validity studies. The candidate's responses to these tests help construct a profile that reflects her/his likely performance level and achievement potential in the job role

This report has the following sections:

The **Summary** section provides an overall snapshot of the candidate's performance. It includes a graphical representation of the test scores and the subsection scores.

The **Insights** section provides detailed feedback on the candidate's performance in each of the tests. The descriptive feedback includes the competency definitions, the topics covered in the test, and a note on the level of the candidate's performance.

The **Response** section captures the response provided by the candidate. This section includes only those tests that require a subjective input from the candidate and are scored based on artificial intelligence and machine learning.

The **Learning Resources** section provides online and offline resources to improve the candidate's knowledge, abilities, and skills in the different areas on which s/he was evaluated.

### Score Interpretation

All the test scores are on a scale of 0-100. All the tests except personality and behavioural evaluation provide absolute scores. The personality and behavioural tests provide a norm-referenced score and hence, are percentile scores. Throughout the report, the colour codes used are as follows:

- Scores between 67 and 100
- Scores between 33 and 67
- Scores between 0 and 33

## 2 | Insights

### English Comprehension

 63 / 100CEFR: **B2**

This test aims to measure your vocabulary, grammar and reading comprehension skills.

You have a good understanding of commonly used grammatical constructs. You are able to read and understand articles, reports and letters/emails related to your day-to-day work. The ability to read, understand and interpret business-related documents is essential in most jobs, especially the ones that involve research, technical reading and content writing.

### Logical Ability

 61 / 100

#### Inductive Reasoning

 58 / 100

This competency aims to measure your ability to synthesize information and derive conclusions.

You are able to work out rules based on specific information and solve general work problems using these rules. This skill is required in data-driven research jobs where one needs to formulate new rules based on variable trends.



#### Deductive Reasoning

 65 / 100

This competency aims to measure your ability to learn by example, imitation or hit-and-trial and provides an indication of how creative the individual is.

It is commendable that you are able to break down complex data to develop specific rules and make logical deductions. You are able to take into account all possible variations to establish valid cause and effect relationships.

### Quantitative Ability

 79 / 100

This test aims to measure your ability to solve mathematical problems on number system, ratio and proportion, interest, profit and loss, percentages, speed, time and distance.

You have a strong hold on basic arithmetic and concepts of algebra. You are able to convert real-world problems into equations and solve them.

### Critical Reasoning

 58 / 100

This test aims to measure your ability to comprehend qualitative information, evaluate logical arguments and arrive at sound decisions.

You are able to reason critically and in an informed manner, as is required in day to day work life. Such critical reasoning skill is relevant for jobs in research and business analysis.

## Managerial In-basket Simulation

50 / 100



### Effective Communication

57 / 100

Effective communication is the ability to be able to clearly convey a message. It is obtained by a combination of use of correct language, and comprehending or listening in a manner that gains the full meaning of what's being said.

It is important to be able to communicate effectively to ensure clear exchange of information without any misunderstandings. This is important in both personal and work relationships as it helps build trust and respect, and improve teamwork and problem solving.

- You may sacrifice the complexity of the information in the message for the sake of brevity, thereby losing important facts and details.
- You may not support your stance with reason and clarification.

#### Suggestion to Improve

- While communicating, you should make sure that you listen to the conversation engagingly and pay equal attention to non-verbal communication.
- You should be more assertive while you communicate as it gathers higher attention from the listener. This helps in making the conversation effective.
- You should read articles and blogs about communicating clearly at a workplace - revert to queries, send a status update, assign work etc.



### Planning and Scheduling

0 / 100

Planning and scheduling is the process of analyzing workplace tasks in order to devise a plan to ensure smooth and timely execution. It involves breaking a task into smaller milestones, prioritizing them and assigning them to various stakeholders. While planning defines WHAT and HOW, scheduling defines WHEN and WHO.

Planning a project provides a roadmap to the developments needed. It helps enlist various tasks and milestones, which help track the progress of a project. A project plan in place makes it easier to assign tasks to stakeholders and estimate deadlines.

- You are unable to think and plan from both long term and short term perspective.
- This may also hinder your ability to utilize the financial, human and informational resources efficiently to meet the necessary project outcomes.
- You exhibit poor understanding of stakeholder interaction and is unable to coordinate effectively with different stakeholders.

#### Suggestion to Improve

- You should analyse the project by breaking it down into multiple phases of implementation and list down the key stakeholders who would be involved in each phase.
- You should have brainstorming sessions where you run your idea with the stakeholders to have a clear achievable action plan after giving due consideration to the suggestions and objections posed by others.
- You should chalk out and schedule your short term plans such that they help in achieving your long term goals as well.



## Process Adherence

56 / 100

Process adherence refers to a situation when a person is compliant to existing processes and operating procedures to ensure delivery of desired outcomes.

The importance of complying with processes and procedures is to ensure efficiency and define accountability in a system. Non-compliance to processes can result in delay in meeting deadlines of a project and may impact quality of the work delivered.

- You try to grasp and understand the processes and steps required to complete a particular task.
- Though you may find it difficult to comply where the processes become slightly more complex to follow through.
- You may try to execute the guidelines on prompting but may find it difficult to keep a track of the guidelines on their own.

### Suggestion to Improve

- You should try to keep abreast with the new processes, and clarify your doubts wherever you find the processes to be complex or difficult to comprehend.
- You should try to identify where the bottlenecks are, and implement corrective actions to prevent recurrence.
- You should create checklists and associate them with the activity of interest - this can help ensure that you are complying with the regulations of a process.
- You should have automated notifications for ticking off process checklists and due tasks list.



## Prioritization

71 / 100

Prioritizing is the process of determining relative importance of various action items and putting them in order of importance.

It is the fundamental basis of planning as it helps recognize relative importance of multiple tasks that may be a part of a project. Knowing the importance and urgency of a task helps determine the order in which they should be performed, and can help move the work closer to long term goals.

- You have a good grasp on prioritizing tasks on both strategic and operational level, so as to ensure actual deliverables.
- You are able to effectively balance various tasks of different urgencies by keeping the communications clear to meet deadlines.

### Suggestion to Improve

- You should work on projects that involve multiple stakeholders and need exceptional work prioritization, and this can help further strengthen your skills.

- You should be flexible while working on projects, so as to be prepared for work that will come above your set order of priorities.



## Work Allocation

68 / 100

- You are likely to be considerate and sensitive to the needs of others.
- You tend to put the needs of others ahead of your own.
- You are likely to trust others easily without doubting their intentions.
- You are compassionate and may be strongly affected by the plight of both friends and strangers.
- You are humble and modest and prefer not to talk about personal accomplishments.
- Your personality is more suitable for jobs demanding cooperation among employees.
- You seem to be comfortable delegating work to team members.
- You understand interdependence of the tasks and you are able to judge who can better handle a given task from among your team members.
- Overall, you are capable of making sound decisions without supervision and possess the requisite skills to delegate tasks responsibly.

### Suggestion to Improve

- You can consider taking ownership of bigger projects and drafting the end-to-end execution plan along with allocation choices which can be reviewed by your supervisor.

## C++ Programming



50 / 100

This test aims to measure the knowledge of programming in the C++ language and the ability to use the C++ standard library to write code.

- You have a strong foundation and a fair understanding of basic algorithms, data structures and object-oriented programming in C++ Programming.
- You have the required knowledge to design and write simple codes.
- You can improve your performance by understanding different types of data structures and the ways to use them while coding.
- You should focus on advanced concepts of C++ Programming and try to develop complex programs.

### 3 | Response

#### Automata



100 / 100

[Code Replay](#)


#### Question 1 (Language: C++20)

You are playing an online game. In the game a list of N numbers is given. The player has to arrange the numbers so that all the odd numbers of the list come after the even numbers.

Write an algorithm to arrange the given list such that all the odd numbers of the list come after the even numbers.


#### Scores

##### Programming Practices

 **100** / 100

High readability, high on program structure. The source code is readable and does not consist of any significant redundant/improper coding constructs.

##### Functional Correctness

 **100** / 100

Functionally correct source code. Passes all the test cases in the test suite for a given problem.

#### Final Code Submitted

Compilation Status: Pass

```
1 // Sample code to read input and write output:
2
3 /*
4 #include <iostream>
5
6 using namespace std;
7
8 int main()
9 {
10     char name[20];
11     cin >> name;           // Read input from STDIN
12     cout << "Hello " << name; // Write output to STDOUT
13     return 0;
14 }
15 */
16
17 // Warning: Printing unwanted or ill-formatted data to output will cause the test cases to fail
18
19 #include <iostream>
20 #include <vector>
21
22 using namespace std;
23
```

#### Code Analysis

##### Average-case Time Complexity

**Candidate code:** Complexity is reported only when the code is correct and it passes all the basic and advanced test cases.

**Best case code:**  $O(N)$

\*N represents size of the list.

##### Errors/Warnings

There are no errors in the candidate's code.

##### Structural Vulnerabilities and Errors

##### Readability & Language Best Practices

Line 27: Variables are given very short name.



```

24 int main()
25 {
26     // Write your code here
27     int n;
28     cin>>n;
29     vector<int>v(n);
30     // int v[n];
31     for(int i=0;i<n;i++){
32         cin>>v[i];
33     }
34     vector<int>ans;
35     vector<int>odd;
36     for(int i=0;i<n;i++){
37         if(v[i]%2==0){
38             ans.push_back(v[i]);
39         }
40         else{
41             odd.push_back(v[i]);
42         }
43     }
44     for(int i=0;i<odd.size();i++){
45         ans.push_back(odd[i]);
46     }
47 // ans.push_back(odd.begin(),odd.end());
48     for(int i=0;i<n;i++){
49         cout<<ans[i]<<" ";
50     }
51
52     return 0;
53 }

```

### Test Case Execution

Passed TC: 100%

Total score



12/12

**100%**

Basic(5/5)

**100%**

Advance(6/6)

**100%**

Edge(1/1)

### Compilation Statistics

6

Total attempts

4

Successful

2

Compilation errors

0

Sample failed

0

Timed out

1

Runtime errors

Response time:

00:08:46

Average time taken between two compile attempts:

00:01:28

Average test case pass percentage per compile:

51.39%

## Average-case Time Complexity

Average Case Time Complexity is the order of performance of the algorithm given a random set of inputs. This complexity is measured here using the Big-O asymptotic notation. This is the complexity detected by empirically fitting a curve to the run-time for different input sizes to the given code. It has been benchmarked across problems.

## Test Case Execution

There are three types of test-cases for every coding problem:

**Basic:** The basic test-cases demonstrate the primary logic of the problem. They include the most common and obvious cases that an average candidate would consider while coding. They do not include those cases that need extra checks to be placed in the logic.

**Advanced:** The advanced test-cases contain pathological input conditions that would attempt to break the codes which have incorrect/semi-correct implementations of the correct logic or incorrect/semi-correct formulation of the logic.

**Edge:** The edge test-cases specifically confirm whether the code runs successfully even under extreme conditions of the domain of inputs and that all possible cases are covered by the code

## Question 2 (Language: C++20)

You are given a list of integers and an integer  $K$ . Write an algorithm to find the number of elements in the list that are strictly less than  $K$ .

### Scores

#### Programming Practices



High readability, high on program structure. The source code is readable and does not consist of any significant redundant/improper coding constructs.

#### Functional Correctness



Functionally correct source code. Passes all the test cases in the test suite for a given problem.

### Final Code Submitted

Compilation Status: Pass

```
1 // Sample code to read input and write output:
2
3 /*
4 #include <iostream>
5
6 using namespace std;
7
8 int main()
9 {
10     char name[20];
11     cin >> name;           // Read input from STDIN
12     cout << "Hello " << name; // Write output to STDOUT
```

### Code Analysis

#### Average-case Time Complexity

**Candidate code:** Complexity is reported only when the code is correct and it passes all the basic and advanced test cases.

**Best case code:**  $O(N)$

\*N represents number of elements in the input list

#### Errors/Warnings

```

13 return 0;
14 }
15 */
16
17 // Warning: Printing unwanted or ill-formatted data to output will c
ause the test cases to fail
18
19 #include <iostream>
20 #include <vector>
21
22 using namespace std;
23
24 int main()
25 {
26     // Write your code here
27     int element_size;
28     cin>>element_size;
29     vector<int>element(element_size);
30     for(int i=0;i<element_size;i++){
31         cin>>element[i];
32     }
33     int num;
34     cin>>num;
35     int answer=0;
36     for(int i=0;i<element_size;i++){
37         if(element[i]<num) answer++;
38     }
39     cout<<answer;
40     return 0;
41 }

```

There are no errors in the candidate's code.

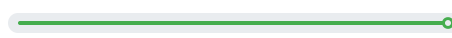
#### Structural Vulnerabilites and Errors

There are no errors in the candidate's code.

#### Test Case Execution

Passed TC: 100%

Total score

 16/16

**100%**

Basic(6/6)

**100%**

Advance(8/8)

**100%**

Edge(2/2)

#### Compilation Statistics

1

Total attempts

1

Successful

0

Compilation errors

0

Sample failed

0

Timed out

0

Runtime errors

Response time:

00:03:11

Average time taken between two compile attempts:

00:03:11

Average test case pass percentage per compile:

100%

### Average-case Time Complexity

Average Case Time Complexity is the order of performance of the algorithm given a random set of inputs. This complexity is measured here using the Big-O asymptotic notation. This is the complexity detected by empirically fitting a curve to the run-time for different input sizes to the given code. It has been benchmarked across problems.

### Test Case Execution

There are three types of test-cases for every coding problem:

**Basic:** The basic test-cases demonstrate the primary logic of the problem. They include the most common and obvious cases that an average candidate would consider while coding. They do not include those cases that need extra checks to be placed in the logic.

**Advanced:** The advanced test-cases contain pathological input conditions that would attempt to break the codes which have incorrect/semi-correct implementations of the correct logic or incorrect/semi-correct formulation of the logic.

**Edge:** The edge test-cases specifically confirm whether the code runs successfully even under extreme conditions of the domain of inputs and that all possible cases are covered by the code

## 4 | Learning Resources

### English Comprehension

[Learn about how to get better at reading](#)



[Read opinions to improve your comprehension](#)



[Improve your vocabulary of terms used in business](#)



### Logical Ability

[Practice inductive reasoning on visual data](#)



[Learn about pattern identification](#)



[Practice your Inductive Reasoning Skills!](#)



### Quantitative Ability

[Learn about the real life applications of logarithms](#)



[Learn about the application of Bayes' Theorem in varied fields](#)



[Learn about Fermet's last theorem](#)



### Managerial In-basket Simulation

#### Effective Communication

[Improve your vocabulary of terms used in business](#)



#### Planning and Scheduling

[Video on How to Schedule Resources on Multiple Projects](#)



#### Process Adherence

[Learn what to do when no one follows your processes](#)



## Prioritization

[List of some Project management tools that can be used to organize tasks](#)



## Work Allocation

[Read how an effective team works](#)



## Icon Index



Free Tutorial



Paid Tutorial



Youtube Video



Web Source



Wikipedia



Text Tutorial



Video Tutorial



Google Playstore